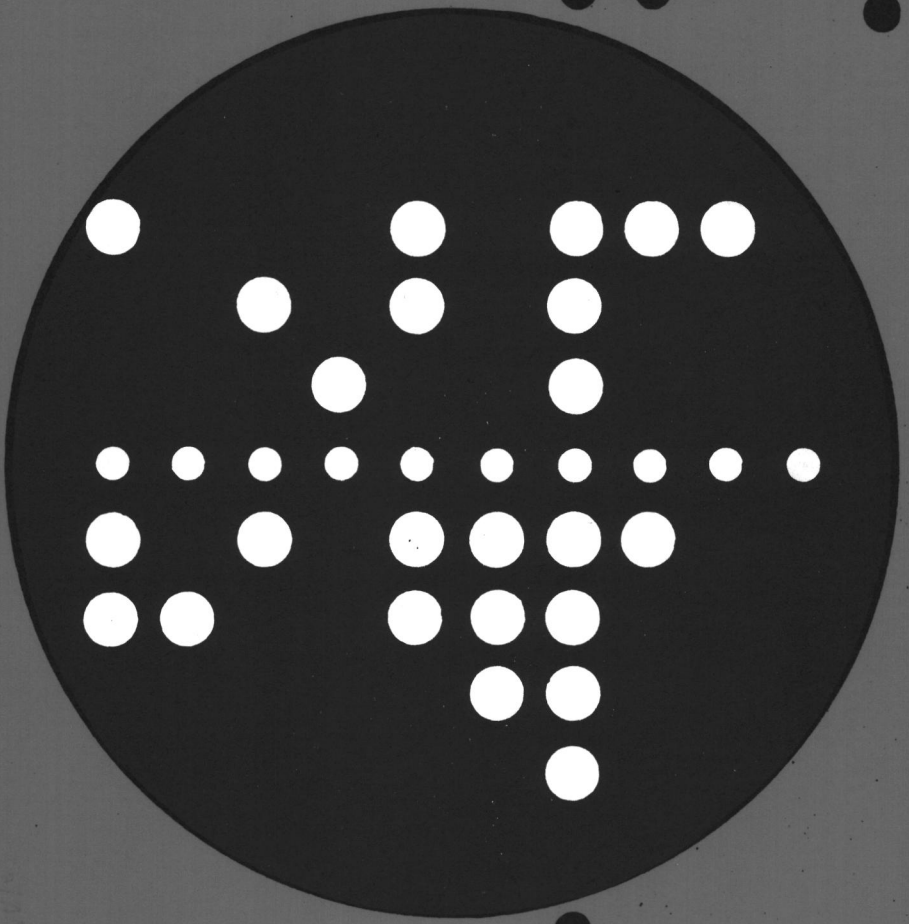
Commission of the European Communities

Joint Research Centre - Ispra

Computing Centre Newsletter



September 1976 • No 4

Contents

Note of the Editor

The present Newsletter will be published monthly except for August and December.

The Newsletter will include:

- Developments, changes, uses of installations
- Announcements, news and abstracts on initiatives and accomplishments.

The Editor thanks in advance those who will want to contribute to the Newsletter by sending articles in English or French to one of the following persons of the Editorial Board.

Note de la Rédaction

Le présent Bulletin sera publié mensuellement excepté durant les mois d'août et décembre.

Le Bulletin traitera des:

- Développements, changements et emploi des installations
- Avis, nouvelles et résumés concernant les initiatives et les réalisations.

La Rédaction remercie d'avance ceux qui voudront bien contribuer au Bulletin en envoyant des articles en anglais ou français à l'un des membres du Comité de Rédaction.

Editor : Sylvia R. Gabbai
Layout : Paul De Hoe
Graphical and Printing Workshop, JRC Ispra

Acknowledgement should be given for their technical support to Mr. E. Eiselt, Mrs. M.G. Giaretta, Mrs. M. Van Andel, Mr. G. Clivio, A. Margnini, G. Zurlo

STRUCTURED PROGRAMMING

C.L. van den Muyzenberg

Program testing is frequently used to convince the programmer of the correctness of his program.

There are some serious drawbacks however:

1. Program testing is time consuming: Programmers in large software projects typically spend their time as follows:
   45 - 50% in program checkout
   30 - 35% in program design
   < 20% in writing the program (1)
2. The bugs still left after testing will produce an enormous amount of lost time and work for the user: IBM's operating system OS/360 contained hundreds of errors when it was released. Two years after issuing RELEASE 18, the **ratio of maintenance costs to development costs was more than 3:1 (2)**

The previous remarks were about large programming projects but remain true for small programs as well; bugs are not partial to large programs.

There are two remarks that point to a different way of establishing the correctness of a program:

"Testing of programs can only demonstrate the presence of bugs, not the absence" (E.W. Dijkstra)

and

"Programs do not acquire bugs by hanging around buggy programs. They acquire bugs only by having programmers insert them" (H. Mills)

The method to write a correct program is not to write first a faulty program and to try to correct it; but to write a correct program from the start. This seems pretty obvious but let's try to refine this a bit by establishing some general rules.

ANALYSIS : PROCEDURE OPTIONS (MAIN)

"Get the complete specifications of the problem";
"check if the preceding statement is executed correctly";

```
DO WHILE ("part of the resulting program remains to be refined");
   "select a part to be refined"
   IF "the selected part can be coded into a programming language"
      THEN DO;
      "write the part as a sequence of programming language
      statements";
      "check that these statements are equivalent to the
      original part";
      END;
      ELSE DO;
      "refine the part using one of the standard control structures";
      "check that the refined text is equivalent to the original part";
      END;
   END;
END ANALYSIS;
```

This meta-program tries to give a description of the top-down structured programming process:

top-down . . starting with the description, refining until the program is written in some programming language

structured . . using standard control structures in the refining process.

Some rules (perhaps obvious) are :

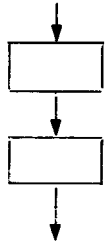
1. get the specifications
2. check the specifications
3. do a refining process until everything is in the programming language you are using,
4. check each refinement if it is equivalent to the original part,
5. when defining variables pay attention to mode, precision and a consistent use of those variables; declare **all** variables,
6. use as long as possible your own language,
7. use standard control structures,
8. use comments and indentation,
9. write a readable program, do not use tricks to save a statement.

As a programming language, every language that contains the standard control structures (or is extendable) could be used. Fortran programmers could use SHELTRAN presented 18-10-1975 by Pollicini. A description of some FORTRAN applications, with the use of SHELTRAN language will be given in one of the next numbers of the Newsletter. Cobol, Assembler and PL/I programmers will find methods to implement standard control structures (2).

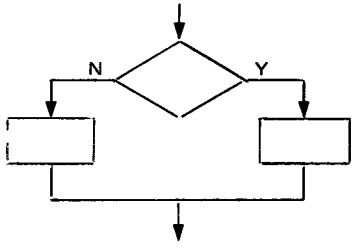
The examples given here will be in structured PL/I (as described in (2)).

The standard control structures have as common property a single entry and a single exit; they are **closed** control structures:

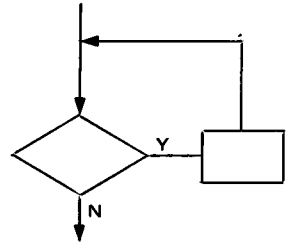
1. SEQUENCE, executing statements sequentially,
2. IF-THEN-ELSE, selection of the next statement depending on a test,
3. DOWHILE, conditional iteration,



SEQUENCE



IF—THEN—ELSE

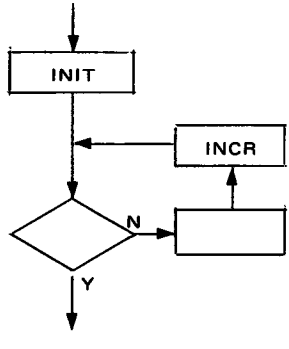


DO—WHILE

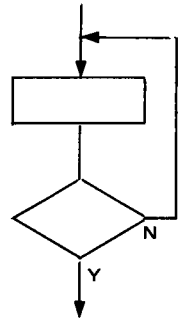
These control structures permit writing any program without the GOTO statement (3); resulting in more easy to read programs and a greater ease in writing them as well.

Although these control structures are sufficient, it is worthwhile to introduce some additional closed control structures:

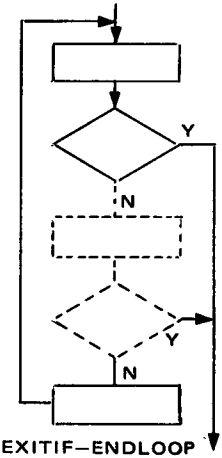
4. ITERATIVE-DO, iteration with incrementation of an index,
5. REPEAT-UNTIL, conditional iteration (similar to DO-WHILE) but the test is made **after** execution of the statements.
6. LOOP-EXITIF-ENDLOOP, a generalization of both DO-WHILE and REPEAT-UNTIL.
7. SELECT-CASE, a generalization of the IF statement, selection of one of several statement groups.



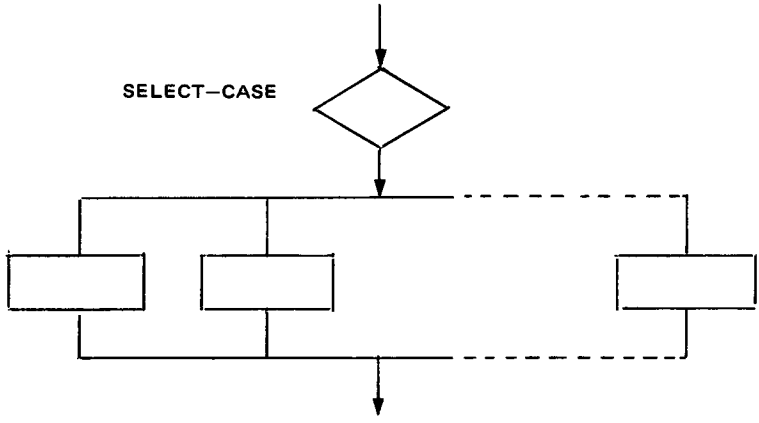
ITERATIVE—DO



REPEAT—UNTIL



LOOP—EXITIF—ENDLOOP



Control structures 1-4 are existing in PL/I, the numbers 5-7 are implemented using the compile-time facilities.

A description of the control structure syntax is given in Appendix A.

JCL Statements

In order to use these control structures at the CETIS, it is necessary to include a DD statement to define the library where the appropriate PL/I compile time procedures are included:

```
//    EXEC   PLPCLG
// CMP.SYSLIB DD  DSN=SYS1.LACVDM,DISP=SHR,
                  UNIT=3330, VOL=SER=USER01.
```

meaning: execute PL/I program product compile, link and go
define a dataset with DD-name SYSLIB for the use by the compiler

PL/I Statements

Before each external procedure (each separately compiled procedure) put a PROCESS statement with the options M (macro) and IS (insource, input list) followed by a statement %INCLUDE STRPLI; to include the compile time procedures.

```
*PROCESS M,IS;
 %INCLUDE STRPLI;        (structures PL/I)
```

The PL/I procedure follows after the include statement. An example of a complete deck and the resulting output is given in Appendix B.

Appendix A

1. SEQUENCE
   syntax [constru | group] ...
2. IF-THEN-ELSE
   syntax IF logex THEN group ELSE group
3. DO-WHILE
   syntax DO-WHILE (logex); sequence END;
4. ITERATIVE-DO (full PL/I syntax for the DO statement)
   syntax DO-statement sequence END;
5. REPEAT-UNTIL
   syntax REPEAT; sequence UNTIL (logex);
6. LOOP-EXITIF-ENDLOOP
   syntax LOOP; { sequence | EXITIF(logex); } ... ENDLOOP;
7. SELECT-CASE
   syntax SELECT (iexp);
   [{ CASE ({ ic | (ic [, ic]. . .) }):} ...
   sequence]
   [OTHER: sequence]ENDSELECT;

with
- constru = any of the 7 control structures,
- group = DO-group or statement or any of the control structures 3-7
- statement = executable statement, not a control structure, not { PROCEDURE | BEGIN | DO END }
- logex = a logical expression (condition)
- iexp = an expression truncated to an integer declared as FIXED BINARY (15)
- ic = an integer constant > 0
- ... = repetition of the preceding unit
- [ ] = optional unit
- { } = unit appears at least once
- | = either the preceding or the following unit

Appendix B

Example (of the use of control structures 5-7)

```
$      CLASS  1
//  EXEC PLPCLG
//CMP.SYSLIB  DD  DSN=SYS1.MACVDM,DISP=SHR,UNIT=3300,
//      VOL=SER=USER01
*PROCESS  M,IS;
 %INCLUDE STRPLI;
 AA:PROC OPTIONS(MAIN);
     I=1;
     REPEAT;
         PUT SKIP DATA(I);
         I=I + 1;
         UNTIL(I>9);
     J=0;
     LOOP;
         J=J +2;
         PUT SKIP DATA(J);
         EXITIF(J>10);
         J=J-1;
         PUT DATA(J);
         ENDLOOP;
     DO K=-2 TO 10;
         SELECT(K);
            CASE(1): PUT SKIP EDIT('CASE 1,   K=',K)(A);
            CASE(8): PUT SKIP EDIT('CASE 8,   K= ',K)(A);
            CASE((4,2,7)): PUT SKIP EDIT('CASE 4,2,7   K=',K)(A);
            CASE(6): PUT SKIP EDIT( 'CASE 6, K= ',K)(A);
            OTHER:PUT SKIP EDIT( 'OTHER,    K= ',K)(A);
            ENDSELECT;
        END;
     END  AA;
```

The program will produce the following output:

```
I  =          1
I  =          2
I  =          3
I  =          4
I  =          5
I  =          6
I  =          7
I  =          8
I  =          9
```

```
J =  2                    J =  1
J =  3                    J =  2
J =  4                    J =  3
J =  5                    J =  4
J =  6                    J =  5
J =  7                    J =  6
J =  8                    J =  7
J =  9                    J =  8
J =  10                   J =  9
J =  11
OTHER, K =  - 2
OTHER, K =  - 1
OTHER, K =  0
CASE 1, K =  1
CASE 4,2,7  K =  2
OTHER  K =  3
CASE 4,2,7  K =  4
OTHER, K =  5
CASE 6  K =  6
CASE 4,2,7  K =  7
CASE 8  K =  8
OTHER, K =  9
OTHER, K =  10
```

The Newsletter is available at:

Mrs R. Porta
Program's Library
Bldg. 36 - Tel. 760

Des exemplaires du Bulletin sont disponibles chez:

Mme R. Porta
Bibliothèque des Programmes
Bât. 36 - Tel. 760

PROGRAMMING SUPPORT NOTE

Copying (part of) a partitioned data set to another partitioned data set

As the SYS1.LIBPROG will not be sustained any more, the users are asked to copy their data set to the USERnn disks.
The data sets must already exist (to create and register a partitioned data set see **Newsletter,** June 1972, No. 2, page 7).

Using the default parameters

the procedure will copy from data set SYS1.LIBPROG on EUSYS2 to a data set on USER01.

Example

```
//        EXEC   COPYPDS,DSOUT='SYS1.MYLIB'
//GO.SYSIN      DD    *
   COPY  OUTDD=DDOUT,INDD=DDIN
   SELECT MEMBER=MEM1
   SELECT MEMBER=(MEM2,MEM3)
```

This will copy the members MEM1, MEM2 and MEM3 from data set SYS1.LIBPROG on EUSYS2 to data set SYS1.MYLIB on USER01.

COPY	The copy statement is fixed. If no SELECT or EXCLUDE statement is given, the whole data set will be copied.
SELECT	Only the members written in a SELECT statement will be copied. More than 1 statement is permitted.
EXCLUDE	All members will be copied except those in an EXCLUDE statement. More than 1 statement is permitted.
MEMBER	may specify 1 member (MEMBER = member name) or more between parentheses (MEMBER = (memb1, memb2,...))

Using all parameters

The parameters are:

DSIN	data-set name input,	default = SYS1.LIBPROG
DSOUT	data-set name output,	default = ***DSOUT (must be specified)
VOLIN	volume input,	default = EUSYS2
VOLOUT	volume in output,	default = USER01

Example

```
//    EXEC COPYDPS,DSIN='SYS1.MYLIB',VOLIN=USER01,
//    DSOUT='SYS1.YOURLIB',VOLOUT=USER02
//GO.SYSIN  DD  *
    COPY   OUTDD=DDOUT,INDD=DDIN
    EXCLUDE MEMBER=(P1,P2,P3,P4)
```

This will copy data set SYS1.MYLIB on USER01 to data set SYS1.YOURLIB on USER02
except for the members P1, P2, P3 and P4, that are not copied.

The procedure invokes the IBM utility program IEBCOPY. For further information on the possibilities of this program, see:

IBM system /360 operating system utilities, GC28-6586

For further information please contact

C.L. van den Muyzenberg

The Computing Centre welcomes the following new staff-members who have taken up their duties over the Summer 1976 :

Mr. D. König,	**responsible for the basic software group**
Mr. J.D. Guinoiseau,	**console-operator**
Mr. W. Bertato,	**operator.**

Statistics of computing installation utilization

Report of computing installation exploitation for the month of June

	YEAR 1976	YEAR 1975
Number of working days	18 d	20 d
Work hours from to for	16.00 h	12.00 h
Duration of scheduled maintenance	24.17 h	19.34 h
Duration of unexpected maintenance	3.83 h	4.03 h
Total maintenance time	28.00 h	23.27 h
Total exploitation time	262.00 h	216.63 h
CPU time in problem mode	128.86 h	85.84 h
Teleprocessing:		
CPU time	1,18 h	0.98 h
I/O number	218.400 h	567.000 h
Equivalent time	2.71 h	4.9 h
Elapsed time	133.50 h	96 h
Batch processing:		
Number of jobs	8.056	8.095
Number of cards read	2.614.000	2.392.000
Number of cards punched	177.000	220.000
Number of lines printed	23.617.000	23.757.000
Number of pages printed	524.000	553.000

BATCH PROCESSING DISTRIBUTION BY CLASS

	A	1	2	3	4	5	D	TOTAL
Number of jobs	1041	2662	1361	1578	348	152	371	7515
Elapsed time (hrs)	25	127	123	160	56	80	69	640
CPU time (hrs)	0.7	21	23	30	16	28	10	128.7
Equivalent time (hrs)	8.7	49.3	49.6	65.4	23.9	42.7	36.6	276.2
Turn around time (hrs)	0.43	0.75	1.3	1.2	3.7	4.2	2.5	1.2

PERCENTAGE OF JOBS FINISHED IN LESS THAN

TIME	15′	30′	1^h	2^h	4^h	8^h	1^D	2^D	3^D	6^D
% year 1975	28.0	16.1	65.1	81.1	89.4	91.4	99.2	99.5	99.5	100
% year 1976	40.7	58.5	72.6	86.1	94.8	97.6	99.0	99.4	99.4	100

Utilization of the computer center by the objectives and appropriation accounts for the month of June

		IBM 370/165 equivalent time in hours
120	General Infrastructure	63,8104
130	Scientific and Technical Support	0.5406
143	ESSOR Reactor	6.9430
145	Medium Activity Laboratory	—
146	Central Bureau for Nuclear Measurements (CBNM)	—
191	Technical Support to Commission Activities	1.8145
193	Technical Support to Power Stations	4.8137
211	Waste Disposal	0.1217
213	Materials Science and Basic Research on Materials	10.1407
214	Hydrogen	1.9410
221	Reactor Safety	54.0903
222	Applied Informatics	25.6705
223	Information Analysis Services	50.8782
230	European Informatics Network	2.6229
251	Standards and Reference Materials	1.6080
252	Protection of the Environment	11.6979
253	Remote Sensing of Earth's Resources	11.1389
254	New Technologies	—
412	Fissile Materials Control	—
	TOTAL	248.3761
190	Services to external Users	31.6403
	TOTAL	**280.0164**

Statistics of computing installation utilization

Report of computing installation exploitation for the month of July

	YEAR 1976	YEAR 1975
Number of working days	22 d	23 d
Work hours from to for	14.00 h	12.00 h
Duration of scheduled maintenance	18.59 h	13.99 h
Duration of unexpected maintenance	39.65 h	3.34 h
Total maintenance time	58.24 h	17.33 h
Total exploitation time	293.76 h	246.67 h
CPU time in problem mode	110.00 h	105.68 h
Teleprocessing:		
CPU time	1.14 h	1.09 h
I/O number	216.000	662.000
Equivalent time	2.66 h	5.72 h
Elapsed time	143 h	115 h
Batch processing:		
Number of jobs	8,105	8,691
Number of cards read	2487,000	2601,000
Number of cards punched	205,000	179,000
Number of lines printed	24832,000	28835,000
Number of pages printed	572,000	649,000

BATCH PROCESSING DISTRIBUTION BY CLASS

	A	1	2	3	4	5	D	TOTAL
Number of jobs	1223	2498	1199	1553	527	70	428	7498
Elapsed time (hrs)	19	100	106	140	82	27	59	533
CPU time (hrs)	0.6	13.6	19.3	24.4	30.9	10.6	8.7	108
Equivalent time (hrs)	6.8	44.4	43.8	63.9	42.3	16.9	34.4	252.5
Turn around time (hrs)	0.4	0.6	1.0	0.9	2.0	2.4	1.7	0.9

PERCENTAGE OF JOBS FINISHED IN LESS THAN

TIME	15'	30'	1^h	2^h	4^h	8^h	1^D	2^D	3^D	6^D
% year 1975	24.4	41.6	60.1	76.6	86.3	88.9	99.4	99.5	99.5	100
% year 1976	43.7	64.2	79.7	90.9	96.9	98.2	99.6	99.8	99.9	100

Utilization of the computer center by the objectives and appropriation accounts for the month of July

IBM 370/165
equivalent time in hours

120	General Infrastructure	72,5397
130	Scientific and Technical Support	0.4431
143	ESSOR Reactor	7.8588
145	Medium Activity Laboratory	—
146	Central Bureau for Nuclear Measurements (CBNM)	—
191	Technical Support to Commission Activities	1.7240
193	Technical Support to Power Stations	1.9727
211	Waste Disposal	0.8145
213	Materials Science and Basic Research on Materials	8.4659
214	Hydrogen	0.5005
221	Reactor Safety	53.7703
222	Applied Informatics	32.1262
223	Information Analysis Services	43.9152
230	European Informatics Network	1.4352
251	Standards and Reference Materials	3.5505
252	Protection of the Environment	15.3389
253	Remote Sensing of Earth's Resources	3.6873
254	New Technologies	—
412	Fissile Materials Control	0.4061
	TOTAL	248.5489
190	Services to external Users	14.0952
	TOTAL	**262.6441**

EQUIVALENT TIME TABLE FOR ALL JOBS OF THE ADMINISTRATION – MONTHLY AND CUMULATIVE STATISTICS

	January	February	March	April	May	June	July	August	September	October	November	December
Year 1975	64	55	62	73	62	61	94	52	51	59	74	70
accumulation	64	119	181	254	316	377	471	523	574	633	707	777
Year 1976	84	82	101	77	57	64	73					
accumulation	84	166	267	344	401	465	538					

EQUIVALENT TIME TABLE FOR THE JOBS OF ALL THE OBJECTIVES – MONTHLY AND CUMULATIVE STATISTICS

	January	February	March	April	May	June	July	August	September	October	November	December
Year 1975	178	171	168	166	142	166	228	137	152	170	190	176
accumulation	178	349	517	683	825	991	1219	1356	1508	1678	1868	2044
Year 1976	206	237	270	241	229	248	249					
accumulation	206	443	713	954	1183	1431	1680					

EQUIVALENT TIME TABLE FOR THE JOBS OF THE EXTERNAL USERS – MONTHLY AND CUMULATIVE STATISTICS

	January	February	March	April	May	June	July	August	September	October	November	December
Year 1975	16	28	24	28	32	31	26	15	18	19	12	18
accumulation	16	44	68	96	128	159	185	200	218	237	249	267
Year 1976	18	19	28	16	25	32	14					
accumulation	18	37	65	81	106	138	152					

EQUIVALENT TIME TABLE FOR ALL JOBS OF ALL USERS – MONTHLY AND CUMULATIVE STATISTICS

	January	February	March	April	May	June	July	August	September	October	November	December
Year 1975	214	216	208	215	190	222	266	166	181	202	219	208
accumulation	214	430	638	853	1043	1265	1531	1697	1878	2080	2299	2507
Year 1976	233	271	313	280	277	281	260					
accumulation	233	504	817	1097	1374	1655	1915					

Acquisition, Manipulation et Stockage de l'Information

J. Pire

Introduction

Le problème de l'introduction des données, dans les systèmes automatiques de traitement de l'information, s'est posé dès le début de la mécanographie.

Les premiers moyens mécanographiques n'ayant pas de mémoires (ou très peu), les fichiers étaient conservés sur leur support de création (cartes et bandes perforées) et chaque traitement impliquait une relecture de ces supports et de nombreuses opérations manuelles avec tous les risques d'erreur inhérents à ces opérations.

Avec les techniques actuelles il faut distinguer nettement entre acquisition des données ou transcription sur un support lisible par ordinateur, et manipulation, c'est-à-dire, mise à jour et mise en archives des fichiers. Il arrive encore malheureusement que certains utilisateurs de centres informatiques modernes en soient restés au concept unique de la carte perforée et sentent encore le besoin de lire de leur propres yeux le contenu du support physique.

Acquisition de l'Information

Nous désignons sous ces termes la transition de l'information sous une forme accessible à un ordinateur.

L'introduction sur le marché des **lecteurs optiques** et la propagande faite autour de ces machines a fait croire que le problème de la transcription était résolu et que les ordinateurs étaient en mesure de lire n'importe quel document. Malgré les énormes progrès faits dans ce domaine observons que les lecteurs optiques sont encore:

- extrêmement coûteux
- exigeants concernant la qualité du document présenté
- limités dans les types de caractères qu'ils sont capables de reconnaître
- exigeants concernant les interventions manuelles.

Bref, les documents lisibles doivent avoir été préparés spécialement pour ce type d'accès à l'ordinateur et ce généralement par des moyens mécanographiques. Les lecteurs optiques sont extrêmement utiles dans certaines applications de type bancaire, réservations de place, termes de magasin, etc., mais il est trop tôt encore pour en espérer un usage généralisé.

A l'heure actuelle, le document original, imprimé ou manuscrit doit être, par l'intermédaire d'un clavier, soit manuscrit sur un support lisible, soit communiqué directement à l'ordinateur.

La **carte perforée** a encore un large emploi comme premier support.
Les quelques avantages qui lui restent actuellement peuvent être décrit comme suit:

- le contnu est généralement interprété, c'est-à-dire, présenté sous 2 formes: une accessible à l'ordinateur et l'autre à l'être humain,
- la carte est relativement robuste.

Les désavantages qui semblent passer inaperçus à bien des utilisateurs sont:

- l'encombrement: c'est probablement le support qui contient le moins d'information par rapport à son poids et ses dimensions
- le coût du support
- les précautions à prendre pour les manipulations
- les précautions nécessaires pour un stockage de longue durée.

Le **ruban perforé** a été le concurrent principal de la carte et est encore largement utilisé actuellement. Les avantages par rapport à la carte sont d'être moins encombrant, et du point de vue fichier, de garantir le respect de l'ordre des enregistrements.

Les desavantages sont:

- sa fragilité relative
- sa manipulation difficile.

En outre les lecteurs de ruban sont généralement peu couteux mais par contre les vitesses de lecture et la fiabilité sont également bien inférieures à celles des lecteurs de cartes.
Pratiquement l'usage du ruban est limité aux petits ordinateurs.

Les minicassettes ont fait leur apparition sur le marché depuis plusieurs années et réalisent la miniaturisation d'un support utilisé depuis longtemps: la **bande magnétique.**

Ses avantages sont incontestables:

- grande densité d'information, faible poids, faible encombrement
- bonne protection du contenu par le boitier
- support réutilisable.

Le désavantage actuel est que les gros ordinateurs ne sont généralement pas équipés pour lire ce support, mais leur emploi se généralise sur les terminaux de téléinformatique et sur les petits ordinateurs pouvant servir de terminaux.

Le **disque souple** (floppy disk) est le dernier né des concurrents; parfaitement adapté aux petits ordinateurs, sa densité d'information est

encore supérieure à celle des minicassettes et l' "accès direct" présente également des avantages par rapport à l'accès séquentiel des bandes.
Le stockage est aisé, la protection est bonne et il est réutilisable.

L'entrée directe par **terminal** à clavier supprime complètement le support externe: les données sont mémorisées directement dans les mémoires périphériques de l'ordinateur où elles peuvent être utilisées et modifiées sans autres interventions manuelles que l'introduction de commandes à un logiciel "ad hoc".

Manipulation d'un fichier

Pendant longtemps la seule façon d'examiner le contenu d'un fichier était de lire l'interprétation du contenu des cartes ou de "lister" ces cartes.

Il est clair que le second procédé du point de vue de l'intégrité du fichier est plus sûr que le premier car il requiert moins de manipulations.
La mise à jour d'un fichier de cartes est hazardeuses car elle remet toujours en question la séquence correcte des cartes.
Les fichiers sur bandes de paper et minicassettes doivent être complètement réécrits (recopiés) pour être modifiés, ce qui peut être assez long, tandis que les mémoires à accès direct peuvent être modifiées sans déplacements inutiles des textes inaltérés. Lorsque l'information est emmagasinée sur un support à accès direct il est toujours possible d'obtenir des sélectives de parties du fichier sans devoir le parcourir complètement, ce qui constitue généralement un gros gain de temps. Les modifications et mises à jour peuvent être introduites soit par terminal soit par un petit nombre de cartes contenant uniquement les instructions de modification et les textes de remplacement.
L'impression des modifications apportées s'obtient généralement automatiquement et les listes nouvelles si nécessaire, sur simple commande.
Un avantage supplémentaire est acquis du fait que le même fichier peut être utilisé par nombreux utilisateurs sans qu'il soit besoin à chacun d'eux de posséder une copie.

Mise en archives

Les mémoires à accès direct sont relativement chères pour mettre en archives des fichiers mais coûte ***moins cher*** que la mise en archives sur cartes.
Le prix d'un disque contenant 100 millions de caractères est de l'ordre de 500.000 lires, il contient dans les plus mauvaises conditions plus de 1 million de cartes, dont le prix actuel est de plus de 2 millions de lires sans compter les meubles nécessaires à leur classement. Le support le moins

cher reste cependant la bande magnétique qui supporte plus de 25 millions de caractères sous un volume restraint pour un prix de l'ordre de 10.000 lires.

Cartes produites par l'ordinateur

Il est évident par là que sortir des résultats sur cartes perforées est une opération coûteuse à tout point de vue: non suelement ces cartes coûtent cher et doivent encore être interprétées, mais les manipulations sont lourdes, malaisées et peu sûres et peuvent même conduire à la destruction de l'information produite. Les résultats en question peuvent d'une part être imprimés et d'autre part être enregistrés temporairement sur disques. Ils pourront être, si nécessaire, mis à jour et réutilisés sans autres interventions que quelques commandes à passer à l'ordinateur, puis soit sauvés sur bandes magnétiques soit simplement détruits.

Facilités disponibles actuellement

Mémorisation des données de travail

Actuellement un espace équivalent à 100 millions de caractères est occupé par plus de 1000 fichiers du type P.S.Q., accessibles tant en mode batch qu'en mode conversationnel à partir d'une quinzaine de terminaux.

Nous constatons malheureusement que certains utilisateurs laissent séjourner dans ces fichiers à accès facile, des textes dont l'utilité est douteuse, étant donné qu'ils restent inutilisés pendant trois mois.

Par ailleurs depuis le 19 juillet 1976, des espaces (près de 200 millions d'octets) sont disponibles sur les disques USER02, USER03, USER04 et USER05 pour mémorisation de fichiers personnels.

Nous avons précédemment publié les listes des procédures d'accès à ces facilités. Nous espérons par là voir progressivement diminuer la circulation des cartes perforées et disparaître les problèmes relatifs à leur interprétation, reproduction, conservation et manipulation en général.

L'unique préoccupation de l'utilisateur est de faire le back-up sur bande magnétique lorsque le contenu du fichier a subi des modifications importantes.

Pour tout renseignement à ce sujet, Messieurs De Wolde (tel. 753) et Van den Muyzenberg (tel. 781) peuvent être consultés.

USING COMPUTERCARDS IS WRONG

H.I. de Wolde

As explained in the article of Mr. Pire, the extensive use of computercards is wrong, dangerous, old fashioned and bears many perils.

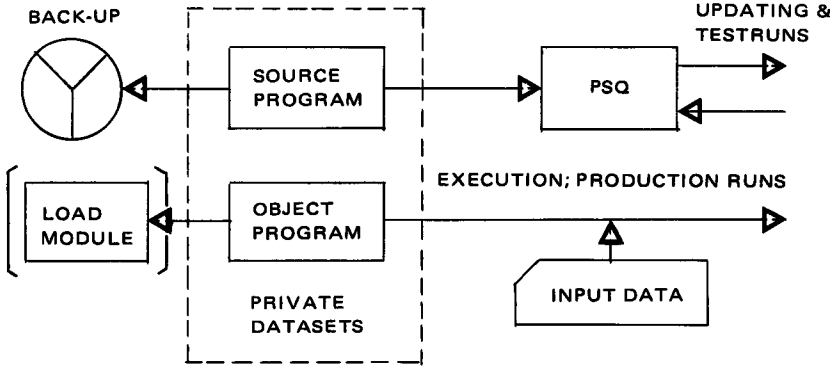
Now we will dedicate some space to the many options which are available to substitute the carddecks.

The basic solution to the programmer is building his own libraries, where he may store source deck, data input and other collections. The hardware tools are magnetic tapes and magnetic disks.
If you choose for using tapes, you may organize your library as you want, without any interference of the system people, which is a big advantage. However, updating and handling is not so easy in this case and your tape(s) must be mounted for every execution. Storage on the available disks(USER01,...,USER05), is flexible and offers the advantage of powerful software tools as PSQ, Librarian, Filedit and IBM utilities.

PSQ itself is a library system with updating facilities. However, as the available space is limited, we should prefer if the programmers would create their own libraries and using PSQ only at the active programming phase. In the production stage one may work directly from the private library.

Operations on a PSQ dataset may be performed in batch processing and in teleprocessing, using the FILEDIT system. Both modes offer the possibilities of updating, inserting and deleding single records or groups of records of your PSQ dataset. The PSQ/FILEDIT manual is available at the Program Library, Mrs. Porta, building 36, tel. 760.

The next diagram gives a possible configuration for your software materials, followed by the appropriate procedures.



1. CREATION AND REGISTRATION OF PRIVATE SPACE

```
//STEP1       EXEC PGM=IEFBR14
//SYSPRINT    DD   SYSOUT=A
//DD1         DD   DSN=dsname,UNIT=3330,VOL=SER=USER0n,
//             DISP=(NEW,KEEP,DELETE),SPACE=(TRK,(i,j)),
//             DCB=(RECFM=FB,LRECL=80,BLKSIZE=800)
//STEP2       EXEC EURUDR,U=3330,V=USER0n
//GO.SYSIN    DD   *
            reservation card
/*
```

in which	dsname	is the name of your private dataset
	n	indicates a diskpack
	i	is the number of primary tracks
	j	is the number of secondary tracks

The necessary space for your dataset may be calculated realising that one track of a 3330 disk may contain up to about 160 cardimages. Of course some space must remain available for updating.
The reservation card is composed as follows:

columns	1 - 7	the number of the "fiche d'activité"
	9 - 12	number of authorization
	14 - 17	number of programmer
	19 - 24	expiration date
	26 - 69	the dataset name, left adjusted
	8, 13, 18, 25	must be left blank

2. LOAD A PROGRAM TO PSQ

```
//STEP1       EXEC PSQ
//GO.SYSIN    DD *
$GENERATE,pgname
$G,xxx
$NY
$PY
            source deck
$E
/*
```

in which	pgname	is the name chosen by you, of up to eight characters, to identifyyour PSQ dataset.
	xxx	are the characters to be punched in columns 73 - 75

The job output will supply you with the listing of the program and the calling number of the PSQ dataset, which have to be combined with the name for sequential processing.

3. TRANSFER OF A PSQ DATASET TO PRIVATE DATASET

```
//STEP1                 EXEC PSQ
//GO.FTO1FOO1           DD   DSN=dsname,UNIT=3330,
//                      VOL=SER=USER0n,DISP=(OLD,KEEP)
//GO.SYSIN              DD   *
$OPEN,yyyypgname
$TO=01
$FM
$E
/*
```

in which yyyy is the PSQ number as supplied by the system.

Take care that you dispose of a good back-up copy of the previous private dataset, because it will be substituted by the new version and consequently the old version will be destroyed.

4. DELETE A PSQ DATASET

Once your updating and testruns are finished and the programs arrived at the production phase, you may cancel your PSQ module as follows:

```
//STEP1                 EXEC PSQ
//GO.SYSIN              DD *
$PURGE,yyyypgname
/*
```

5. TRANSFER OF A PRIVATE DATASET TO THE PSQ SYSTEM

```
//STEP1                 EXEC PSQ
//GO.FT01F001           DD   DSN=dsname,UNIT=3330,
//                      VOL=SER=USER0n,DISP=(OLD,KEEP)
//GO.SYSIN              DD *
$GENERATE,pgname
$NY
$PY
$TI=01
$FM
$E
/*
```

6. COMPILATION OF A PROGRAM AND LOADING OBJECTDECK TO RESERVED SPACE

This procedure may be used to initialize the production phase of a program. One must first reserve the necessary space for the object deck, as is shown in procedure 1. The source deck and the object deck are two independent datasets.

```
//STEP1          EXEC FTGC
//CMP.SYSLIN     DD   DSN=obname,UNIT=3330,
//               VOL=SER=USER0n,DISP=(OLD,KEEP)
//CMP.SYSIN      DD   DSN=pgname,UNIT=3330,
                      VOL=SER=USER0n,DISP=(OLD,KEEP)
```

in which obname is the dataset name of the object deck.

The execution of the program is performed through:

```
//STEP1          EXEC FTLG
//LKED.SYSIN     DD   DSN=obname,UNIT=3330,
//               VOL=SER=USER0n,DISP=(OLD,KEEP)
//GO.SYSIN       DD *
                      input data
/*
```

7. BACK-UP OF SOURCE PROGRAM

Many programmers suffered nervous breakdowns, because of negligence in producing regular back-up tapes. Please do create back-up copies of your programs and refresh them regularly. Do keep a good administration including all the modifications since the last back-up. The two next procedures describe the actions for respectively a PSQ dataset and a private datset. For both of them you must dispose of a labeled tape.

7A. PSQ DATASET TO TAPE

```
//STEP1          EXEC PSQ
//GO.FT01F001    DD   UNIT=TP9,VOL=(PRIVATE,SER=zzzz),
//               DSN=dsname,LABEL=(1,SL,,OUT),DISP=(NEW,PASS),
//               DCB=(RECFM=FB,LRECL=80,BLKSIZE=800)
//GO.SYSIN       DD *
$OPEN,yyyypgname
$TO=01
$FM
$E
/*
```

in which zzzz is your tape number

7B. PRIVATE DATASET TO TAPE

```
//STEP1          EXEC   PGM=IEHMOVE
//SYSPRINT       DD     SYSOUT=A
//SYSUT1         DD     UNIT=SYSDA,DISP=(NEW,DELETE),
//                      SPACE=(CYL,(1,1))
//DD1            DD     DSN=dsname,UNIT=3330,
//                      VOL=SER=USER0n,DISP=(OLD,KEEP)
//DD2            DD     UNIT=TP9,VOL=(PRIVATE,SER=zzzz),
//                      DSN=dsname,LABEL=(1,SL),DISP=(OLD,PASS),
//                      DCB=(RECFM=FB,LRECL=80,BLKSIZE=800)
//SYSIN          DD *
     COPY  DSNAME=dsname,TO=TP9=zzzz,                              C
                FROM=3330=USER0n,TODD=DD2
/*
```

Note : Character C must be punched in column 72.

8. RESTORE A DATASET FROM BACK-UP TAPE

If your private datset has been ruined or destroyed, you may restore from your back-up tape through procedure 7B, with the same control statements but modifying the data input as follows:

```
     COPY  DSNAME=dsname,FROM=TP9=zzzz,                            C
                TO=3330=USER0n,FROMDD=DD2
```

The receiving dataset must be newly created as the utility IEHMOVE refuses to overwrite existing datasets.

9. LARGE QUANTITIES OF INPUT DATA

Ij your problem requests large quantities of input data, you may of course use the described procedures for data storage also. The data input description becomes in this case:

```
//GO.SYSIN       DD   DSN=dtname,UNIT=3330,
//                    VOL=SER=USER0n,
//                    DISP=(OLD,KEEP)
```

in which dtname is the datasetname containing the input data.

The here presented procedures are not exhaustive to the problem of reducing card collections. They just offer some solutions. Many variations are possible. For example the program in the production phase may be stored as a load module, as has been described in the Newsletter No. 2 of June 1976.
For any particular problem please contact the support group.

Les personnes intéressées et désireuses de recevoir régulièrement "Computing Centre Newsletter" sont priées de remplir le bulletin suivant et de l'envoyer à:

Mme R. Porta
Bibliothèque des Programmes
Bât. 36, Tel. 760

Nom .....

Adresse .....

.....

Tel. .....

The Persons interested in receiving regularly the "Computing Centre Newsletter" are requested to fill out the following form and to send it to:

Mrs R. Porta
Program Library
Building 36, Tel. 760

Name .....

Address .....

.....

Tel. .....